Modeling, Storing and Mining Moving Object Databases

Sotiris Brakatsoulas   Dieter Pfoser   Nectaria Tryfona

Research Academic Computer Technology Institute
Athens, Hellas
{sbrakats|pfoser|tryfona}@cti.gr

Abstract.

Urban areas get more and more congested everyday due to the increasing number of moving vehicles. This imposes the need for efficient analysis, modeling, and processing of traffic data. Moreover, the extraction of additional information about traffic conditions, optional routes and the possible prediction of troublesome situations, such as traffic jams, becomes necessary. In this work, we describe the analysis, pre-processing, modeling, and storage techniques for trajectory data that constitute a Moving Object Database (MOD). MOD is the backbone of the ΙΧΝΗΛΑΤΗΣ ('PATH-FINDER' in Greek) *system, which specifically focuses on extracting further information about the movement of vehicles in the Athens municipal area. Based on real-world requirements, we initially analyse the traffic data and make modeling decisions to capture these requirements in a MOD. We then design MOD focusing on the spatiotemporal concepts, relations and restrictions among the characteristic concepts of the system – namely, the vehicles, trajectories, and roads. Furthermore, specific, innovative pre-processing, design, and storage techniques for the trajectory data in MOD are given. Then, we present the architecture of ΙΧΝΗΛΑΤΗΣ; its core components are the characteriser, cluster finder, and associator, which are used to perform data extraction in MOD. A mining language to accommodate typical data extraction queries is presented, in terms of syntax and semantics. Answers to characteristic, complex questions on MOD, which are based on real-world data about traffic in the Athens Metropolitan Area, show the applicability of the approach.*

1 Motivation

As the number of moving vehicles increases rapidly everyday, the need for analysis, modeling and processing of traffic data is vital. Moreover, the extraction of additional information about traffic conditions, optional routes and possible prediction of troublesome situations, such as traffic jams, becomes necessary.

In this work, we deal with the analysis, pre-processing, modeling and storage techniques of traffic data in a Moving Object Database (MOD) for a traffic management system. Furthermore, we apply data extraction techniques in MOD, assisting the prediction of difficult situations, or optional cases, such as alternative routes when traffic is congested.

In order to realize MOD, based on real-world requirements about traffic in the Athens Metropolitan Area, we analyse the fundamental concept of the movement of a vehicle and register its semantics and properties in terms of a conceptual model. We organize them in a database, including vehicles, routes, trajectories (i.e., traces left behind as vehicles move), and relations among them.

Building a MOD is not a trivial issue. It consists of (i) spatial data, (i) non-spatial data and (iii) trajectories. Spatial data comprises infrastructure information such as roads, buildings, obstructions, etc. and the non-spatial data consists of other thematic information. Both data scenarios are well-explored research subjects and various commercial DBMS products exist that allow for their efficient manipulation. Trajectory data on the other hand is a rather new field of research and no commercial products are available to manage it. Handling trajectories includes (i) the pre-processing of the data, i.e., dealing with errors in positional measurements, (ii) data modeling, i.e., defining a conceptual data model that meets systems requirements and (iii) data storage, i.e., the logical data model, data types, and query processing issues. In this work, we outline these problems and present approaches for addressing them.

Furthermore, for a MOD to be useful, it is not only enough to register current information but also to extract further knowledge from it. In our case, MOD is the backbone of the ΙΧΝΗΛΑΤΗΣ system [9], which focuses on extracting further information about the movement of vehicles in the Athens municipal area. The core of the ΙΧΝΗΛΑΤΗΣ system are the characteriser, cluster finder and associator components that are built on top of MOD. In order to realize these components, we adapt known data extraction functions that exist in literature, namely, characterization, clustering and association, to the needs of moving object data. We build a Spatial Mining Language to support these functions, in terms of syntax and semantics, and we show how typical, complex data extraction queries are accommodated with the support of the underlying model.

This work is part of a larger research project focusing on the development of methods and techniques to build a traffic management system, namely ΙΧΝΗΛΑΤΗΣ, based on real-world data from the Athens Metropolitan Area. In this paper, we present the preliminary results of this effort. The contribution is twofold: (i) the registration of the semantics of moving object data in an object-oriented way, resulting in a MOD, and (ii) the adaptation of the well-known and widely-used mining functions of characterization, clustering, and association in the moving object application domain and their typical expression through the Spatial Mining Language (SML), allowing their formal application in MOD.

Related work in this area includes mainly commercial tools focusing either on the spatial data mining area, such as the XpertRuleMiner [25], Conquest [26], and GeoMiner [27] [8], or on the traffic/fleet management, such as the E-track [21], Diplomat [22], VFL[23], and Accu-Tracker [24]. The traffic/fleet management applications are focused on the management of current movement information and do not include an analysis component for historic data. The area of data mining for traffic data has focused on building data warehouses and algorithms to visualize the huge amounts of traffic count data stored in traffic and transportation databases [17] [18] [19]. These data is different from the trajectory approach pursued in this work since movement is representing as volume data for discrete spatial locations. Further, various approaches to store and query trajectory data have been proposed in literature, e.g., [11] [14] [20]. However their distinct and decisive disadvantage is that they cannot easily be implemented in practice, i.e., by using and extending off-the-shelf DBMSes to manage the data. We are not aware of any work trying to model at a high level the semantics of traffic data and movement. Previous work includes [12] [13] [15] and [16], focusing independently on the movement and the spatial mining process respectively.

The rest of the paper is organized as follows. Section 2 gives the organization of the database of the ΙΧΝΗΛΑΤΗΣ system. Section 3 deals with important pre-processing, modeling and storage issues related to trajectory data in MOD. Section 4 presents the architecture and components of ΙΧΝΗΛΑΤΗΣ. The mining component, performing the data extraction process has a core role. Section 5 analyses the mining process in the traffic management system and presents the spatial mining language; its syntax and semantics. Characteristic queries show the applicability of the approach. Finally, Section 6 this work.

2 Organizing the Moving Object Database

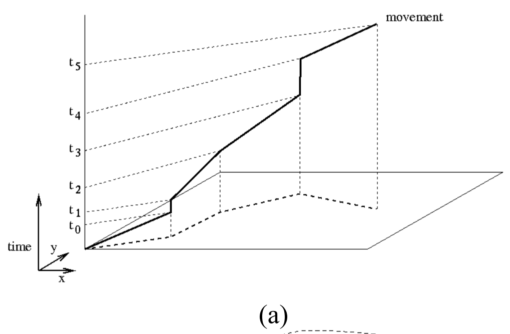
In order to realize and organize a traffic management system, it is essential to understand and study the moving objects, their properties and relations, as well as the fundamental concept of *movement* of objects, which is basic in all application domains whether they deal with moving vehicle or with users carrying a mobile phone. After defining the semantics involved in objects movement and the concepts that need to be captured, we organize them in a database, the Moving Object Database (MOD). MOD is the core of the traffic management system. Later on, pre-processing methods, storage techniques and data extraction functions will be performed on this database.

2.1 The Semantics of Movement

Consider a scenario using a traffic management system to monitor the traffic flow in the city of Athens, Greece. By monitoring the movement of specific vehicles (e.g., delivery trucks, public transport, taxis, etc.) one can ask the following queries: 'find the vehicles that just entered Athens', or 'find the vehicles that left Athens an hour ago,' or more general 'find locations with a larger number of vehicles' (i.e., typical traffic jam pre-condition). Representing such moving objects as point objects (their volume or size does not play a critical role) their movement can be illustrated as shown in Figure 1. The solid line in Figure 1(a) represents the movement of a point object. Space (x- and y-axes) and time (t-axis) are combined to form a 3D-area. The dashed line shows the projection of the movement in two-dimensional space (x and y coordinates).

In order to record the movement of a vehicle, we need its position on a continuous basis. However, GPS and telecommunications technologies only allow us to sample an object's position, i.e., to obtain the position at discrete instances of time such as every few seconds. By, later on, interpolating these samples, we can extract the movement of the object. The simplest approach is to use linear interpolation, as opposed to other methods such as polynomial splines [1]. The sampled positions then become the end points of line segments of polylines, and the movement of an object is represented by an entire polyline in three-dimensional space. In geometrical terms, the movement of an object is termed a *trajectory*; in other words, trajectory is the trace of the vehicle in time.

Figure 1(b) shows a spatiotemporal space (the cube in solid lines) and several trajectories (the solid lines) contained in it. Time moves in the upward direction, and the top of the cube is the time of the most recent position



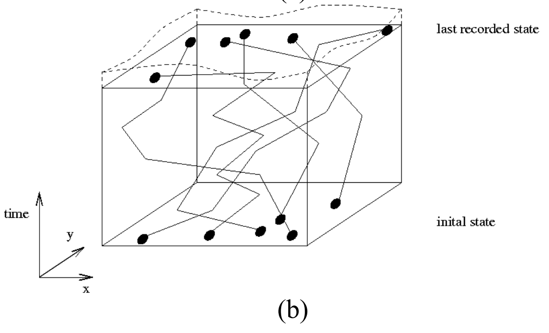


Figure 1: Moving point objects: (a) a trajectory and (b) several trajectories evolving in a finite region

sample. The wavy-dotted lines on top symbolize the growth of the cube with time.

The trajectory representation is adequate to derive certain properties and relationships of the object movement.

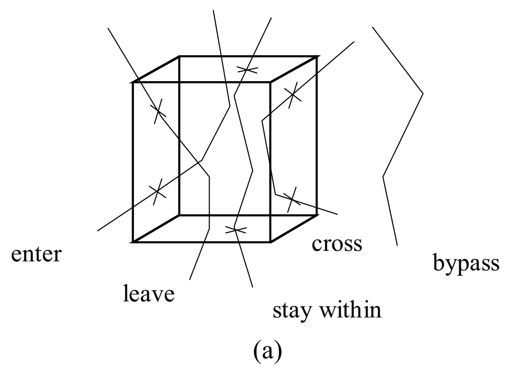
Properties

Trajectories are characterized by a set of different properties depending on the application requirements. The most common properties are: (a) the speed of the movement (b) the heading, showing the direction of the vehicle, (c) the covered area, indicating the area the vehicle covered during its trip, (d) the traveled distance, and (e) the traveled time. Based on our studies [11] [12], the aforementioned representation is adequate for mobile database modeling since it gives answers to simple questions such as 'which area did vehicle A-4592 cover during its trip?' and to more complex ones, like 'which vehicles left Athens after midnight moving East and were found close to each other 2 hours later?'.

Relationships

Through their movement, trajectories relate to their environment in different ways over time. In the following, we discuss to types of relationships, namely how a trajectory can relate to its (spatial) environment and to other trajectories.

Relations between a trajectory and its spatial environment. Trajectories can have relations with other spatial objects. This can be infrastructure elements, such



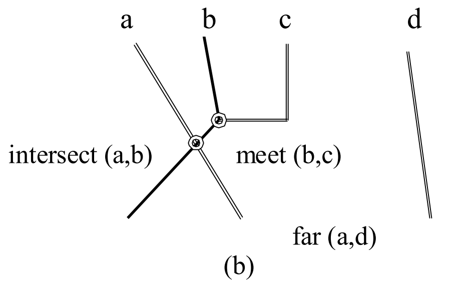


Figure 2: Relationships: (a) trajectory/spatial environment and (b) trajectory/trajectory

as roads, parks, buildings, etc. but also imaginary entities, such as city boundaries or query regions. In the temporal context these spatial entities become three-dimensional (i.e., space and time dimensions) represented by e.g. a 3D region. We distinguish the five basic relationships *stay within*, *bypass, leave, enter, cross* (cf. Figure 2(a)), but others can also be included.

Relations among trajectories: Additionally, relevant positions among trajectories need to be registered at time points. The most common ones based on topological reasoning [4] are the following (Figure 2(b) depicts four of them): *intersect*, *meet*, *equal*, *near*, and *far*.

2.2 The Database Schema of MOD

The various concepts relating to trajectories presented in the previous section need to be organized to define the underlying data model of MOD. We initially use conceptual modeling to capture the semantics of the aforementioned concepts in an organized manner. For the conceptual representation, we use the class diagram of UML [2] due it is popularity, high-degree of comprehension and expressiveness.

Figure 3 illustrates the conceptual schema of MOD. To capture a 'trajectory', we need an identification of the mobile device (indicated by 'object id'), the actual trajectory ('trajectory id') as well as the position of the trajectory itself. In other words, 'position' describes the trace of the moving vehicle. The data types used are abstract, since they only should indicate the dimensionality of the parameter. More concrete instances of data types can be found in, e.g., [7]. A set of

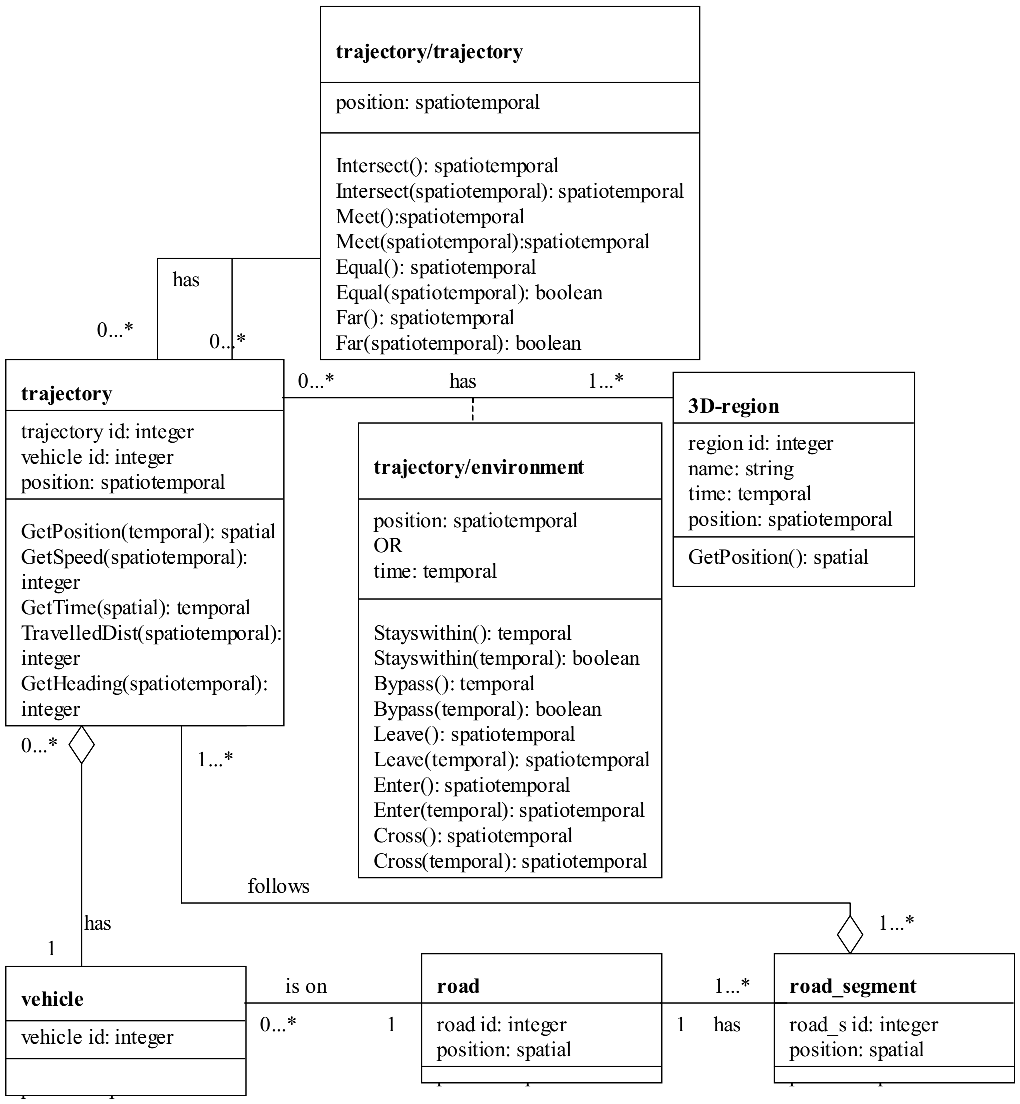


Figure 3: An excerpt of the database schema of MOD

operations, e.g., GetSpeed(spatiotemporal), GetTime(spatial), and TravelledDistance(spatiotemporal), GetHeading(spatiotemporal) are prototypical and show what type of information can be derived from the trajectory data, e.g., to compute the traveled distance or the heading of a trajectory, we apply an operation that uses a spatiotemporal range as a parameter.

The '3D-region' class is prototypical to denote the spatial environment of the trajectory; in this case it shows the total covered area.

Trajectories 'have' (one or more) relations either with other trajectories, or their 3D-region class. Figure 3 contains the respective functions to compute such relationships. E.g., 'Leave' without parameter computes the spatiotemporal positions at which a trajectory left a given instance of a 3D-region class. To restrict the operation, we can use an argument to the function. In the case of Leave it is a temporal argument, i.e., the search for spatiotemporal positions at which the trajectory has left the region is restricted to a given time interval. In the class 'trajectory/environment' the parameter 'position' or 'time' capture the result of the function. Equally, so does 'position' in relation 'trajectory/trajectory'.

The rationale and choices presented here have the main advantage of describing two basic concepts: (a) the trajectory of the moving object by keeping track of its movement, and (b) the moving object itself, by recording its last known position. The spatiotemporal framework in

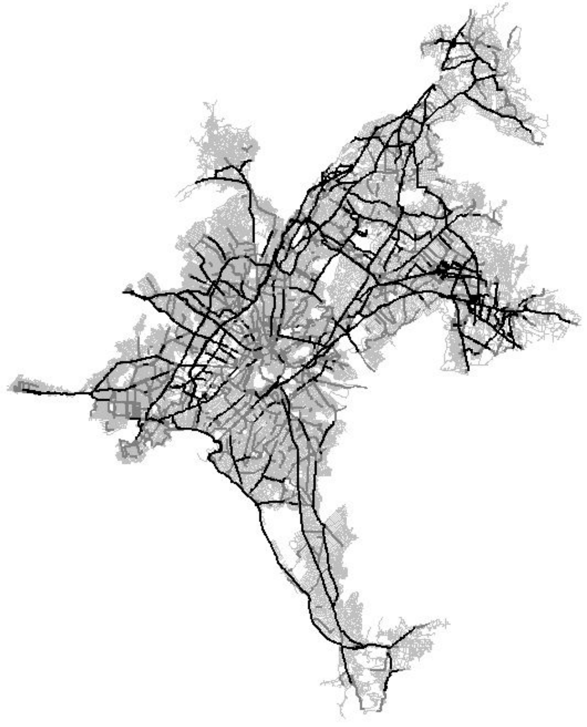


Figure 4: The road network of Athens, Greece (~150k edges)

which the movement takes place can either be built on the fly (i.e., while the objects move) or be pre-defined (e.g., Athens in a specific time interval).

3 Managing Trajectory Data

After having modeled MOD, it is essential to represent and manage the trajectory data. As illustrated in Figure 3, the core components of MOD are the trajectory, and spatial and non-spatial data. Spatial and non-spatial data are well-explored research subjects and various commercial DBMS products exist that provide this functionality. Approaches for trajectory data on the other hand are rare and no commercial products are available to manage this kind of data. However, storing these data is just but one of the many issues related to managing trajectories. Thus, in the following, we outline the problems and detail various solutions relating to (i) pre-processing, (ii) modeling, and (iii) storing these data.

3.1 Map Matching

Trajectory data is obtained by sampling the movement, i.e., by taking positional measurements of the movement at discrete points in time. In traffic management applications, typically GPS is used for this purpose. A positional measurement with GPS is not precise but has an error associated. Although as the technology gets more and more mature the error decreases in size, at current a measurement error of 15 meters is rather common without the use of additional technology such as differential GPS or WAAS (Wide Area Augmentation System). Because of this error, mapping the position of a vehicle onto a road network is not a trivial task, especially if one has to map a trajectory that consists of not only imprecise but also sparse positional measurements.

The technique to map positional measurements onto a map, specifically a road network, is commonly referred to as *map matching*. For this work, we developed an algorithm tailored to the tracking data that was available to us. In our case, the dataset consists of ca. 26000 trajectories that in turn consist of 11 million segments. The data was collected through GPS vehicle tracking in the municipal area of Athens through the years 2000 to 2003. The sampling rate was 30 seconds, i.e., a positional measurement of the movement was taken every 30 seconds. Our map matching algorithm mapped these trajectory data onto a vector road map of the metropolitan area of Athens, Greece. The road network consists of roughly 150 000 edges and extents over an area of 40 x 40 km. Figure 4 shows the Athens road network with some heavily traversed roads indicated in black and dark gray.

Having created the data, one needs to store it. In the following section, we discuss a data model for trajectory data that is based on the data produced by the map matching algorithm.

3.2 A Data Model for Trajectory Data

The map matching algorithm allows us to map the trajectories onto a road network. Essentially, the road network captures the spatial aspect of the trajectories. By exploiting this aspect one can store trajectories more efficiently than by using a “raw” 3D representation. The conceptual schema shown in Figure 5 illustrates the approach of capturing and, later on, storing trajectory data by means of an underlying movement network. It comprises three main components relating to the (i) vehicle information, (ii) trajectories, and (iii) network. In the following, the respective names of entities, relationships, and attributes are indicated in parenthesis. The spatial aspect of the trajectories is modeled in terms of the network, which consists of edges (EDGE) and nodes (NODE). Each edge has a ‘from node’ (FROM NODE) and a ‘to node’ (TO NODE). Additionally, we capture the incident edges of nodes (HAS EDGES). The geometry of a node is a two-dimensional point. The trajectories (TRAJECTORY) are related to the network edges in that a trajectory consists of segments (SEGMENT), which in turn relate to (RELATE TO) network edges. Network edges in conjunction with nodes define the spatial extent of the road network and thus the *spatial aspect* of the trajectories. The *temporal aspect* of the trajectory is captured by assigning two timestamps

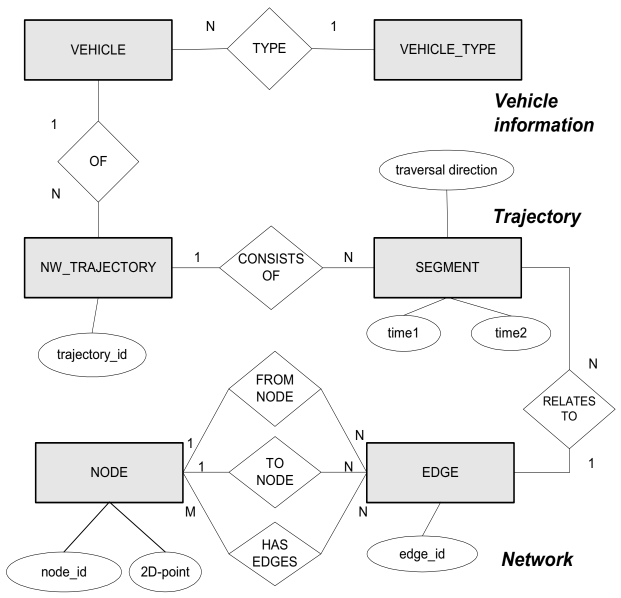


Figure 5: A schema for trajectories in MOD

(time1 and time2) to the segment entity, indicating the start and end times.

Besides the trajectory data, also vehicle information relating to the moving object that "produced" the trajectory is captured. This information becomes useful in subsequent data processing (e.g., routes heavily frequented by delivery trucks).

We should point out that Figure 3 illustrates an excerpt of the database schema of MOD, while Figure 5 depicts the schema for designing and further implementing just the trajectory data. The dotted lines lead to comments.

3.3 Trajectory Data Storage

Using the conceptual schema of the previous section, we store the trajectory in a set of tables using specifically an Oracle DBMS version 9i. We took advantage of the Oracle Spatial functionality by using spatial data types. Our logical schema is defined in terms of four tables.

The table **NW_TRAJECTORY(trajectory_id, edge_id, time1, time2)** records trajectory segments, with the trajectory referenced by its respective id. Each segment represents a unique street edge traversal. Further, we record the id of the street edge the trajectory passes through, the time it enters the edge and the time it leaves the edge. Relating to the conceptual schema of Figure 5, entities TRAJECTORY and SEGMENT are collapsed into this relation. For storing the road network, we use the following three relations.

NODE(node_id, 2D-point) represents the spatial aspect of the street network. Each tuple represents a street node (a junction). The attribute 2D-point is of a geometric point type. The attribute node_id uniquely identifies a node.

EDGE_NODES(edge_id, node_id1, node_id2) allows us to identify the start/end nodes of each network edge.

NODE_EDGES(node_id, edge_id) identifies for each node the incident edges.

To facilitate query processing, we use various indexes. A spatial index is used for the network nodes, a composite B-tree index is used on the edge_id, time1, time2 attributes (NW_TRAJECTORY relation), and on the node_id and edge_id attributes (NODE_EDGES relation). A B-tree index is also created for the node_id attribute (NODE_EDGES relation). Querying the trajectory data typically involves all four relations.

Table 1 summarizes the size of the various relations that are needed to store the 26000 trajectories. In an experimental evaluation of this approach, we also created tables to store the 3D trajectory segments including the creation of a 3D R-tree index. The storage requirement for this approach (for the same trajectory data) was 4.6 GB, thus almost five times as high as for the network approach. This difference is due to the elimination of redundancy in the trajectory data by storing its spatial properties separately in the network.

4 The Architecture of the ΙΧΝΗΛΑΤΗΣ System

After having designed a MOD and captured trajectory data in it, we present the architecture of the ΙΧΝΗΛΑΤΗΣ traffic management system. MOD is its backbone and it contains trajectory data, other spatial data, such as maps or building blocks, and non-spatial data, such as thematic attributes, texts and pictures.

The goal of the system is (a) to tackle efficiently issues related to trajectory data, such as preprocessing, modeling and storage and (b) to act as a mining tool for further knowledge and data extraction and thus to help in

Table 1: Trajectory data storage occupation

Network Schema	
Table or Index	**Size (MB)**
NW_TRAJECTORY	476.41
NW_TRAJECTORY_INDEX	480.2
NODE	5.95
NODE_INDEX	12.53
NODE_EDGES	6.12
NODE_EDGES_INDEX	9.22
EDGE_NODES	5.9
EDGE_NODES_INDEX	3.4
Total	**999.73**

decision making by assisting on the prediction of troublesome traffic situation.

Based on literature [28] [5] [6], for a system like ΙΧΝΗΛΑΤΗΣ, to perform the mining process, it is essential to support: (i) *feature/data extraction*, which focuses on obtaining only the interesting attributes or relations of the data in the database. A typical example of this case is the query: 'find all the equivalent routes, in terms or travel time' or 'find all current trajectories going west', and (ii) *pattern extraction* and *discovery*, which extracts further knowledge about the current situation, based on patterns [28]. For example, 'find the traffic load between 7.00 am-10.00am during the weekdays, and the weekends'.

In our design, according to this, we include the two components of *data extraction* and *pattern extraction*. In this work, we developed, until now, the feature/data extraction component. The data extraction component further includes the modules of *trajectory characterizer*, *trajectory cluster finder* and *trajectory associator*, while the *trajectory patter finder* is responsible for pattern extraction and discovery.

Each component is realized through a mining function, namely, characterization, clustering, association and pattern (Section 5.1), which allow for the appropriate data acquisition. Finally, these mining functions are expressed through the Spatial Mining Language, which captures the semantics of the mining functions in a syntactic way (Section 5.2). Figure 6 depicts the architecture of the ΙΧΝΗΛΑΤΗΣ system.

5 Mining MOD

In this section we present the application of data extraction – as part of the mining process – on MOD, in order to extract further knowledge. By using the concepts of the underlying model of Figures 3 and 5, the data extraction is possible. We first give an informal definition of the data extraction functions and then we give their syntactic and semantic definition.

It is important to point out that in this paper we are concerned *only* with those data extraction (or mining) functions that gain special meaning and importance when combined with positional data; for this, we chose to work with the more basic and widely-used functions. The pattern finder function, corresponding to the pattern extraction component of Figure 6, is a whole research effort of its own, and is not presented in this paper.

5.1 Mining Functions

The role of data mining functions is to query already existing information in order to extract further knowledge and reveal behavioural patterns and trends that are useful in the decision making process. This is usually accomplished by examining data from different perspectives and combining them. In this section we present three fundamental, widely used, data extraction functions [4] [7] [5], namely *characterization*, *clustering*, and *association*. In literature, many more mining function proposals *do* exist. Some of them overlap semantically, for example classification, and characterization, or appear with different names, such as summarization instead of generalization, while others depend exclusively on the application domain such as dependency analysis (i.e., to describe the value of some characteristics based on the value of another characteristic) or deviation detection for business-oriented environments. However, some of the functions appear to be fundamental for the mining process as they are based in common-sense techniques for further information extraction. The presented approach is open to any extension about including other mining functions.

To avoid any confusion, we give the definition of the three well-visited mining operations:

Characterization is the task of assigning a new attribute to a class based on some attribute values. For example, for further analysis, all trajectories moving west from the center of the city show the vehicles (and thus the traffic load) leading to the 'west-suburban areas'.

Clustering creates a new object class based on the values of some attributes. Clustering can be based on spatial information, e.g., 'all trajectories heading west' or non-spatial information, e.g., 'all vehicles with the travelled distance of 20Km between [10-10.30], towards the same direction, are clustered as 'equivalent_routes''.

It becomes obvious that the distinction between clustering and characterization is really thin, as it is also apparent by the confusion of these terms in literature. Generally speaking, it is a matter of a design decision whether we choose to create a new cluster or a new attribute, depending on the size of the result data we are dealing with as well as the semantics and emphasis we want to add to our MOD.

Association of data. This indicates a relationship between object classes, or in other words, the presence of one pattern identifies another pattern. For example, 'the relationship between 'vehicle position' at a certain time point.' More specifically, finding 'all vehicles coming close to each other 'now'', gives a good prediction for a possible traffic jam.

5.2 The Spatial Mining Language (SML) of the ΙΧΝΗΛΑΤΗΣ system

After having designed MOD for the traffic management system, and discussed the three most-used mining functions, we proceed by applying them to the traffic management system context by means of a mining language. The mining functions have the generic syntax:

```
MINE mining function
ON/AMONG object class(-es)
AS composite spatial constraint
```

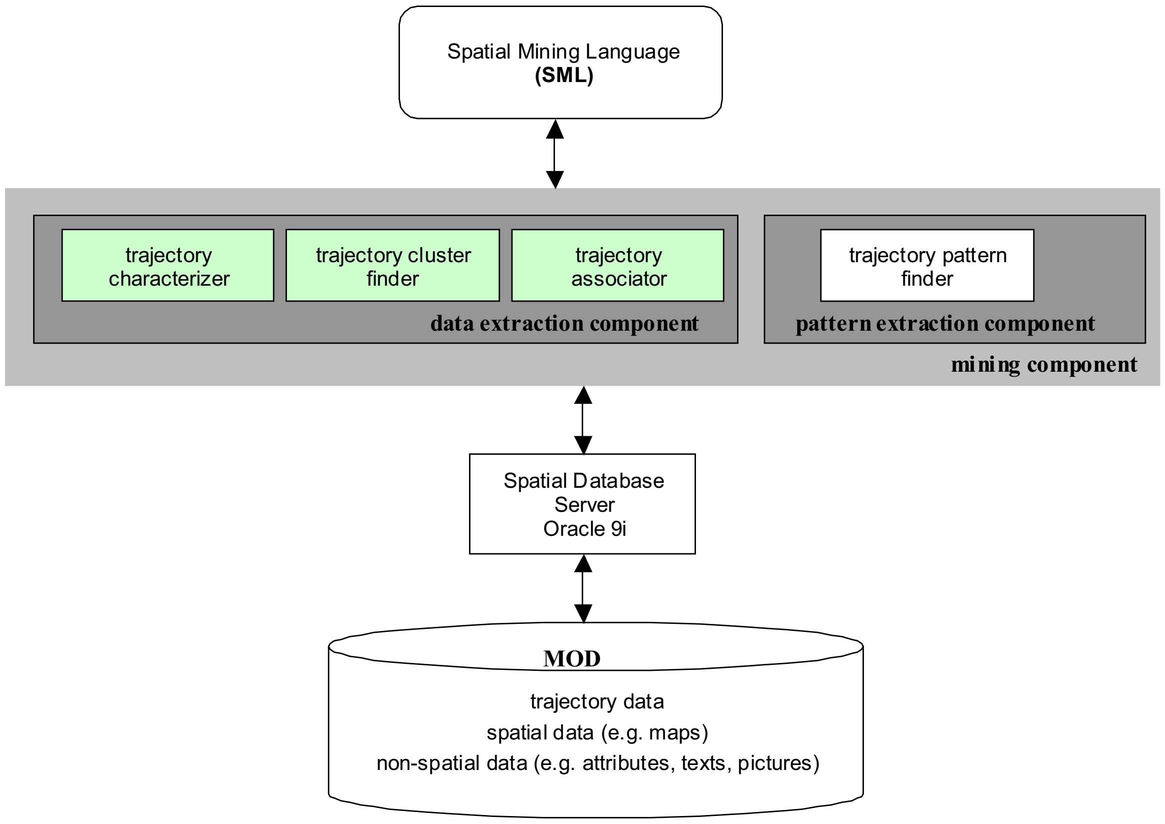


Figure 6: The architecture of the ΙΧΝΗΛΑΤΗΣ system

where *mining function* is one of the 3 fundamental ones, performed on MOD *object classes* and *composite spatial constraint* is an expression built of the basic spatial constraints (i.e., relations among trajectories and among trajectories and 3D environments (c.f. Section 2.1)) using conjunction and disjunction.

Next, we present the formal syntax of the Spatial (data) Mining Language (SML) together with its semantics. SML allows for the definition of characterization, clustering and association, together with spatial constraints; SML is open to extensions to accommodate more mining functions. One can argue that the language can also be used to mine conventional databases and quite understandable so, since the concept of space in captured only in the spatial constraints participating in the **AS** clause.

Upper case words denote reserved words. Lower case words denote variables, standing for arbitrary names. Words in lower case with capitalized initial denote the value of the variable, e.g., *attr* stands for attribute name, while *Attr* stands for its value. Clauses in < > are optional arguments; in (,...) are repeatable; in { ... | ... } are alternatives (one of); in ( ) indicate grouping of arguments.

Characterization adds a new attribute to an object class, based on the values of other attributes.

```
MINE CHARACTERIZATION new_attr_name
ON {object class}
AS {f(obj_class_attr^i_j,…)}
```

where i,j are integers, and $1 \leq i \leq$ (number of object classes), thus $\text{obj\_class\_attr}^i_j$ indicates the j-th attribute of the i-th class. f is a function indicating the composite spatial constraint.

Clustering creates a new object class, based on conditions on attribute values of another class.

```
MINE CLUSTERING {new_obj_class}
ON {obj_class^i}
AS {f(obj_class_attr^i_j,…)}
```

where i, j integers and $1 \leq i \leq$ (number of classes), and $\text{obj\_class\_attr}^i_j$, is the j-th attribute of the i-th object class. The ON clause indicates object class out of which the new class is created.

When CLUSTERING is performed in space we define the result as the geometric union of the position.

Association relates object classes, based on attribute values.

```
MINE ASSOCIATION
AMONG  (obj_class_y,obj_class_z,…)
AS f(attr^m_n,…)
```

where m, n, y, z are integers, obj_class_y, obj_class_z are object classes and $attr^m_n$, with $1 \leq m \leq$ (number of object classes), is the n-th attribute of the m-th class.

5.3 Data Extraction about Traffic

We show examples of its applicability using real-world queries concerning traffic in the area of Athens, Greece. The functions are applied in the MOD of ΙΧΝΗΛΑΤΗΣ traffic system and thus, refer to the object classes, relations, attributes and operations illustrated in the database schema of Figure 3.

Query 1: *Find all vehicles with a travelled distance of 15 to 20 km from the center of Athens towards South, between 10:00 to10:30 and cluster them as 'equivalent_routes'.*

By knowing the equivalent routes in terms of travel time, the system can provide us with alternatives in case of a traffic jam or other troublesome situation in one of these routes.

```
MINE CLUSTERING 'equivalent_routes'
ON trajectory
AS (15 km < distance(GetPosition(10:00) –
GetPosition(10:30)) < 20 km) and (170 <
GetHeading(spatial extent: center±20km,
temporal extent: 10:00 – 10:30) <  190
(degrees))
```

The travelled distance is computed as the Euclidean distance between the position of the moving object at 10:00 and 10:30, respectively. The 'South" direction is determined by the trajectory having a heading between 170 and 190 degrees. The GetHeading function is constrained by a spatiotemporal range constrained by a 20 km from the center (actually up to $\sqrt{2}*20$ since we use a box and not a circle to approximate this range) and the temporal constraint from 10:00 to 10:30 (cf. Figure 7).

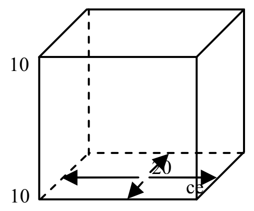


Figure 7: Spatiotemporal range: spatial extent: center ± 20km, temporal extent: 10:00 – 10:30

Query 2: *Find the relative positions of all vehicles that left Athens at 10.00 pm.*

The information implies the traffic in the specific routes the vehicles followed.

```
MINE ASSOCIATION vehicles that left Athens 1
hour ago
AMONG 3D-region trajectory
trajectory/environment
AS (region.name='Athens') ∧
(trajectory/environment. trajectory-
id=trajectory.trajectory-id)
∧ (3D-region. region-
id=trajectory/environment.region-id) ∧
(leave(10.00pm) intersect 3D-region.position)
```

The *intersect* operation corresponds to the intersect spatial relation of [4] and is used to indicate all the routes that left the Athens area at 10.00 (cf. Section 2.2).

Query 3: *Find all vehicles heading 'northeast' from the center and classify them as 'going_to_Kiffisia'.*

This is a characteristic query to extract traffic load in traffic management systems.

```
MINE CHARACTERIZATION 'going to kifissia'
ON trajectory
AS (40 < GetHeading(spatial extent: center ±
2km, temporal extent: 'now - 5min' – 'now') <
60 (degrees))
```

The 'Northeast' direction is determined by the trajectory having a heading between 40 and 60 degrees. The GetHeading function is constrained by a spatiotemporal range constrained by a 2 km from the center (representing the 'center' area) and the temporal constraint from up to 5 minutes ago until now.

6 Conclusions

In order to realize and organize a traffic management system, named ΙΧΝΗΛΑΤΗΣ, it is essential to understand and study the objects, their properties and relations, as well as the fundamental concept of *movement*, which is basic in all application domains dealing with moving vehicles. We, initially, define the semantics involved in objects movement and concepts that need to be captured, we organize them in a database, the Moving Object Database (MOD). Next, we focus on pre-processing methods, and storage techniques for the trajectory data. We used a network-based representation of trajectories that was obtained by applying map-matching techniques. The trajectory data was stored by means of an off-the-shelf DBMS. Then, we define the architecture of the ΙΧΝΗΛΑΤΗΣ system. In order for ΙΧΝΗΛΑΤΗΣ to perform the mining process, it is essential to firstly support feature/data extraction, which focuses on obtaining only the interesting attributes of the data in the database. The data extraction component includes the modules of *trajectory characterizer*, *trajectory cluster finder* and *trajectory associator*, which are expressed through the Spatial Mining Language.

Acknowledgements

Supported by *DB-Globe: A Data-centric Approach to Global Computing* (IST-2001-32645), and *ΙΧΝΗΛΑΤΗΣ: A Traffic Management System* (Hellenic General Secretariat of Research and Technology).

References

[1] Bartels, R.H., Beatty, J.C., and Barsky, B.A., 1987. An Introduction to Splines for Use in Computer Graphics & Geometric Modeling. Morgan Kaufmann Publishers, Inc.

[2] Booch, G., Rumbaugh, J., and Cobson, I., 1999. The Unified Modeling Language User Guide. Addison-Wesley.

[3] Clifford, J., Dyreson, C.E., Isakowitz, T., Jensen, C.S., and Snodgrass, R.T., 1997. On the Semantics of ``Now" in Databases. ACM Trans. Database Syst. 22(2): 171-214

[4] Egenhofer, M.J., 1991. Reasoning about Binary Topological Relations. SSD 1991: 143-160

[5] Ester M., Kriegel H.-P., Sander J., 1999. 'Knowledge Discovery in Spatial Databases', LNCS, Vol. 1701, 1999, pp. 61-74.

[6] Goebel, M., and Gruenwald, L.: A Survey of Data Mining and Knowledge Discovery Software Tools. ACM SIGKDD. June (1999).

[7] Güting, R., Böhlen, M., Erwig, M., Jensen, C. S., Lorentzos, N., Schneider, M., and Vazirgiannis, M.: A Foundation for Representing and Querying Moving Objects. *ACM Transactions on Database Systems* 25(1), pp. 1-42, 2000.

[8] Han, J., Koperski, K., and Stefanovic, N., 1997. GeoMiner: A System Prototype for Spatial Data Mining. ACM-SIGMOD, Tucson, Arizona.

[9] ΙΧΝΗΛΑΤΗΣ. A Traffic Management System, 2002-2004. General Secretariat of Research and Development, Hellas.

[10] Pfoser, D., 2000. Issues in the Management of Moving Point Objects. PhD Dissertation, Aalborg University.

[11] Pfoser, D., Jensen, C.J., and Theodoridis, Y., 2000. 'Novel Approaches in Query processing for Moving Objects'. VLDB, Cairo, Egypt, Sept. 2000.

[12] Pfoser, D., and Theodoridis, Y., 2000. Generating Semantics-Based Trajectories of Moving Objects. International Workshop on Emerging Technologies for Geo-Based Applications, Ascona, Switzerland.

[13] Pitoura, E., Abiteboul, A., Pfoser, D., Samaras, G., and Vazirgiannis, M., 2003: DBGlobe: a service-oriented P2P system for global computing. SIGMOD Record 32(3): 77-82.

[14] Prasad Chakka, V., Adam Everspaugh, A., and Patel, J.M., 2003. Indexing Large Trajectory Data Sets With SETI. CIDR 2003, 1st Biennial Conf. on Innovative Data Systems Research, Asilomar, CA, USA, January 5-8, 2003, Online Proceedings.

[15] Tryfona, N., 2000. A Framework for Constraint-based Spatial Data Mining. Nectaria Tryfona. International Workshop on Emerging Technologies for Geo-Based Applications, Ascona, Switzerland.

[16] Tryfona, N., and Pfoser, D., 2001. Designing Ontologies for Moving Objects Applications. Nectaria Tryfona and Dieter Pfoser. International Workshop on Complex Reasoning on Geographic Data. Paphos, Cyprus.

[17] Shekhar, S., Lu, C.T., Tan, X., Chawla, S., 2001. Map Cube: A Visualization Tool for Spatial Data Warehouses, as Chapter of Geographic Data Mining and Knowledge Discovery. Harvey J. Miller and Jiawei Han (eds.), Taylor and Francis, 2001.

[18] Shekhar, S., Lu, C.T., Zhang, P., and Liu, R., 2002. Data Mining for Selective Visualization of Large Spatial Datasets, 14th IEEE ICTAI, Nov. 2002.

[19] Shekhar, S., Lu, C.T., Chawla, S., and Zhang, P., 2001. Data Mining and Visualization of Twin-Cities Traffic Data, Dept. of Computer Science Technical Report TR 01-015, U. of Minnesota, 2001.

[20] Tao Y., and Papadias, D., 2001. MV3R-Tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries. VLDB. pp. 431-440.

[21] E-track, www.etrackfleet.com. As of January, 2004.

[22] Diplomat, www.dcs.com. As of January, 2004.

[23] VFL, www.onlinedata.gr. As of January, 2004.

[24] AccuTracker, http://universaltracking.com/. As of January, 2004.

[25] XpertRule Miner, http://www.attar.com/. As of January, 2004.

[26] Conquest, http://dml.cs.ucla.edu/projects/oasis/Conquest/conquest.html. As of January, 2004.

[27] GeoMiner, http://db.cs.sfu.ca/GeoMiner/. As of January, 2004.

[28] Ullman, J., Data Mining Lecture Notes, http://www-db.stanford.edu/~ullman/mining/mining.html. As of January 2004.